Information Extraction for Event Ranking[*]

José Devezas[1] and Sérgio Nunes[2]

1 **INESC TEC and Faculdade de Engenharia, Universidade do Porto, Porto, Portugal**
`jld@fe.up.pt`

2 **INESC TEC and Faculdade de Engenharia, Universidade do Porto, Porto, Portugal**
`ssn@fe.up.pt`

Abstract

Search engines are evolving towards richer and stronger semantic approaches, focusing on entity-oriented tasks where knowledge bases have become fundamental. In order to support semantic search, search engines are increasingly reliant on robust information extraction systems. In fact, most modern search engines are already highly dependent on a well-curated knowledge base. Nevertheless, they still lack the ability to effectively and automatically take advantage of multiple heterogeneous data sources. Central tasks include harnessing the information locked within textual content by linking mentioned entities to a knowledge base, or the integration of multiple knowledge bases to answer natural language questions. Combining text and knowledge bases is frequently used to improve search results, but it can also be used for the query-independent ranking of entities like events. In this work, we present a complete information extraction pipeline for the Portuguese language, covering all stages from data acquisition to knowledge base population. We also describe a practical application of the automatically extracted information, to support the ranking of upcoming events displayed in the landing page of an institutional search engine, where space is limited to only three relevant events. We manually annotate a dataset of news, covering event announcements from multiple faculties and organic units of the institution. We then use it to train and evaluate the named entity recognition module of the pipeline. We rank events by taking advantage of identified entities, as well as *partOf* relations, in order to compute an entity popularity score, as well as an entity click score based on implicit feedback from clicks from the institutional search engine. We then combine these two scores with the number of days to the event, obtaining a final ranking for the three most relevant upcoming events.

1998 ACM Subject Classification I.2.7 Natural Language Processing, H.3.3 Information Search and Retrieval

Keywords and phrases Named Entity Recognition, Relation Extraction, Knowledge Base Population, Entity-Based Ranking, Academic Events

Digital Object Identifier 10.4230/OASIcs.SLATE.2017.18

1 Introduction

Semantic search is not yet a reality. Nonetheless search engines are continuously striving to become more semantic-aware [14, 2]. Most modern search engines already rely on a well-

[*] José Devezas is supported by research grant PD/BD/128160/2016, provided by the Portuguese funding agency, Fundação para a Ciência e a Tecnologia (FCT). This work is partially supported by Project "TEC4Growth – Pervasive Intelligence, Enhancers and Proofs of Concept with Industrial Impact/NORTE-01-0145-FEDER-000020", financed by the North Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, and through the European Regional Development Fund (ERDF).

curated knowledge base, having a basic understanding of telegraphic natural language queries. It is query understanding [12] that supports the "decoration" of results with contextual widgets that directly provide answers to the users questions by either displaying lists of entities (e.g., "sci-fi movies from 1985") or information about a single entity (e.g., "back to the future"), usually pointing to related queries (e.g., "Directed by Robert Zemeckis" or "Time travel movies") or even to queries leading to other relevant entities (e.g., "Michael J. Fox" or "Christopher Lloyd", which are part of the cast). Longterm solutions to support this kind of entity-oriented search require increasingly better automatic approaches for the extraction of entities and relations, as well as the improved integration of heterogeneous data sources, such as textual content and multiple knowledge bases that potentially need to be aligned and merged.

There are several challenges that need to be tackled in order to enable the creation of a truly semantic search engine. In this work, however, we cover the basics, by providing an overview of a complete information extraction pipeline, from data acquisition to knowledge base population. We then present an actual application of the extracted information, where we take advantage of identified entities and relations to improve a widget from an entity-oriented institutional search engine. In particular, we focus on a query-independent ranking problem, where the goal is to rank news that cover event announcements, in order to display the three most relevant upcoming events of general interest to the local academic community. This is illustrated in Figure 1, where we show the landing page for the ANT search engine[1], highlighting the upcoming events widget. Our approach to event relevance scoring is based on three factors: (*i*) the overall popularity of the mentioned entities, (*ii*) the overall popularity of the mentioned entities from clicked event news, and (*iii*) the number of days left until the event starts. Despite lacking an innovative facet, we present a complete implementation of an information extraction pipeline, for the Portuguese language, describing the whole process from data acquisition to the final application, where we solve a ranking problem from a real-world search engine with the knowledge base we automatically constructed.

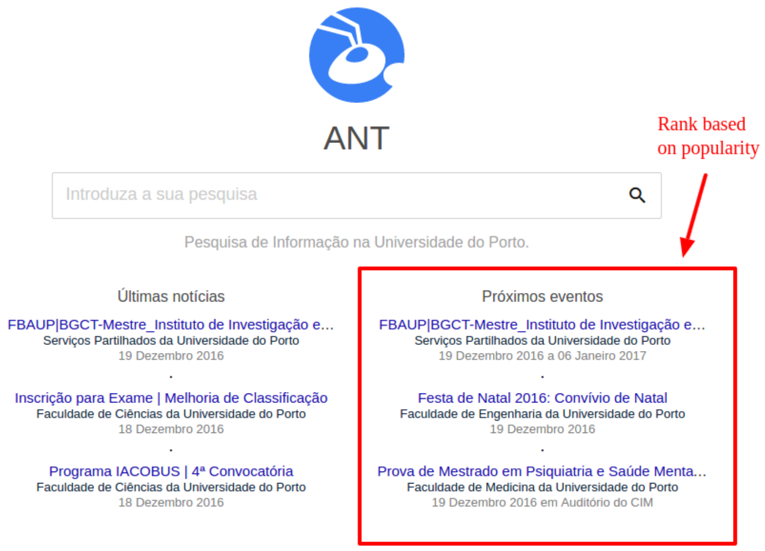
2 Reference Work

In order to better grasp the full scope of the problem we tackle here, we require an understanding of the typical information extraction approaches, as well as available tools and corpora. We also require a bit of context regarding the integration of information extraction (IE) and information retrieval (IR), in particular with a focus on applications to semantic search.

2.1 Information Extraction

Information extraction consists of uncovering information from unstructured data. While it can be applied to different types of data, we only focus on textual content. Natural language expressed as text is an extremely rich source of information and the most common means of sharing knowledge among humans. Machines, however, rely on the identification of structure within this kind of unstructured data in order to be able to find answers to increasingly complex questions. Therein lies the goal of information extraction. Given a text, a typical pipeline would consist of named entity recognition (e.g., "José Devezas" or "InfoLab") and classification (e.g., "Person" or "Organization"), followed by the extraction

[1] `http://ant.fe.up.pt`



ANT

Introduza a sua pesquisa

Pesquisa de Informação na Universidade do Porto.

Últimas notícias

FBAUP|BGCT-Mestre_Instituto de Investigação e...
Serviços Partilhados da Universidade do Porto
19 Dezembro 2016

Inscrição para Exame | Melhoria de Classificação
Faculdade de Ciências da Universidade do Porto
18 Dezembro 2016

Programa IACOBUS | 4ª Convocatória
Faculdade de Ciências da Universidade do Porto
18 Dezembro 2016

Próximos eventos

FBAUP|BGCT-Mestre_Instituto de Investigação e...
Serviços Partilhados da Universidade do Porto
19 Dezembro 2016 a 06 Janeiro 2017

Festa de Natal 2016: Convívio de Natal
Faculdade de Engenharia da Universidade do Porto
19 Dezembro 2016

Prova de Mestrado em Psiquiatria e Saúde Menta...
Faculdade de Medicina da Universidade do Porto
19 Dezembro 2016 em Auditório do CIM

Figure 1 ANT – Entity-Oriented Search Engine for the University of Porto. Upcoming events widget is highlighted, illustrating the target application of the entity-based ranking strategy.

of relations between identified entities, usually based on patterns centered around verbs (e.g., "works at"). So, given a text with the sentence "José Devezas works at InfoLab.", an information extraction pipeline would be able to build a machine-understandable statement for the "[Person] works at [Organization]" relation, identifying "José Devezas" as a "Person", "InfoLab" as an "Organization" and "works at" as the relation between "José Devezas" and "InfoLab". This type of relations is usually stored in a triple store. A triple store contains statements (or triples) in the format *(subject, predicate, object)*, which, as a whole, form a knowledge base. The knowledge base is a semantic network that links concepts and provides a structured way of accessing information – individual statements represent information that, when combined, can provide knowledge.

Information extraction can be classified as closed or open, regarding the domain of application. When the domain is closed, we can easily take advantage of handcrafted rules or gazetteers (dictionaries of entities) in order to link chunks of words to a particular entity, via a regular expression or through string matching, respectively. Open information extraction is used when the domain is unknown or too extensive to cover manually, as is the case of the web. Banko et al. [1] focus on the extraction of relational tuples from the web, without any human input, taking into account issues like automation, corpus heterogeneity and efficiency. They present TextRunner, which uses a self-supervised learner to measure the trustworthiness of candidate tuples, along with a single-pass extractor to identify each candidate tuple, using a redundancy-based assessor module to decide on which trustworthy tuples to keep, with a given probability. Their system is also able to be queried, in a distributed manner, supporting complex relational queries that go beyond a traditional search engine.

The typical result of an information extraction pipeline is a knowledge base. Google is currently supported on their own well-curated knowledge base, Knowledge Graph, which evolved from Freebase. However, while perhaps less well-known, they are also working on Knowledge Vault [8], an automated approach to build the automated equivalent of Knowledge Graph. Knowledge Vault does information fusion based on supervised learning and prior knowledge derived from existing knowledge bases, associating a probability with each extracted statement. Their goal is to enable question answering and entity-oriented search systems, with fewer dependence on manual labor.

After building such a knowledge base, and in order to potentiate semantic search, one of the main relevant techniques is entity linking [22]. First, there is the need to identify entities in a text, for instance with resource to techniques like Conditional Random Fields [13], which is the approach we use in our pipeline. Then, named entity disambiguation is usually required to be able to identify the entity associated with a given word or sequence of words. While we did not implement a disambiguation step within our pipeline, given the relevance of this technique in general and for future work, we still present an interesting example. Nebhi [18] proposed a named entity disambiguation strategy based on entity popularity and syntactic features, trained using a Support-Vector Machine classifier. Entities extracted from text were matched with entities in a knowledge base (in this case Freebase was used) and the disambiguation module determined which candidate entity was more likely to be associated with that particular textual representation. Entity popularity was used as fallback (when everything else fails, it's probably the most frequent), while syntactic features were responsible for providing richer indicators than a bag-of-words approach, being able to capture dependencies between words in a sentence and, in a way, providing additional context.

2.1.1 Tools

There are several tools for natural language processing, implementing named entity recognition, as well as relation extraction. The tool we decided to use was the Natural Language ToolKit (NLTK) [3]. NLTK is a Python library that integrates well with other tools such as MaltParser [19], a dependency parser written in Java, or Stanford NER[2], a named entity recognition Java command line tool and library, based on Conditional Random Fields (CRFs). NLTK directly provides access to corpora in multiple languages, to train for instance a Part-Of-Speech (POS) tagger. POS tagging can be used as a feature for named entity recognition or relation extraction. In this work, we use Stanford NER without POS tags to extract entities, but build regular expressions based on words and their POS tags to extract relations.

In order to learn a CRF for the Portuguese language, able to support a custom set of entity types, we first needed to annotate our own dataset. One of the easiest ways to annotate a dataset is to use the Brat rapid annotation tool [23]. Brat is a web application, built in Python, that requires little configuration and a corpus of plain text documents to annotate. We will provide additional detail on the configuration and usage of Brat in Section 3.

Other relevant tools include LemPORT [21], a lemmatizer for the Portuguese language. Lemmatization can be useful for instance to resolve multiple verbal forms of the same verb into a single word, enabling better relation extraction. This is less efficient but more effective than stemming, where the suffix of the word is trimmed with the goal of obtaining common prefixes for similar words. Also regarding the Portuguese language, a well-known tool is REMBRANDT [4], a framework for semantic information retrieval, trained with the Second HAREM Portuguese corpus [16] and achieving first place in the competition, with an F-measure of 0.625. Finally, while not specifically relevant for the Portuguese language, we also cite GATE [6], a General Architecture for Text Engineering and, in particular Mímir [5], a multiparadigm indexing and retrieval framework, which is capable of searching

[2] `http://nlp.stanford.edu/software/CRF-NER.shtml`

over heterogeneous data sources, like text, annotations, ontologies and knowledge bases. GATE Mímir shares many of the goals of the ANT search engine, combining information extraction and retrieval in a single framework.

2.1.2 Corpora

There are at least two Portuguese corpora, available through NLTK, with annotated POS tags: Mac-Morpho [9] and Floresta [10]. These datasets are also called treebanks, which are corpora annotated with syntactic or semantic sentence structure. In this particular work, we only rely on Floresta treebank to train a POS tagger based on its *(word, tag)* tuples. The POS tags can be useful for named entity recognition and for relation extraction, but we only use this feature during relation extraction to build regular expressions.

While there are several tools for the English language, with pre-trained models for multiple tasks, the Portuguese language has been less explored, despite having its own particular challenges. These include dealing with European and Brazilian Portuguese, or even capturing the changes from the latest Portuguese orthography reform (1990)[3]. More importantly, the lack of resources for the Portuguese language is of great concern. One of the main, or perhaps the only publicly available dataset with annotated entities and relations for the Portuguese language is HAREM [16]. There are two versions of this dataset, but only the second version includes annotations regarding semantic relations between entities. While there are other datasets, their availability is a challenge, as they are held hostage as a valuable resource in a competitive world of science. One way to provide a larger amount of annotated data is simply to push for the semantic web and to motivate people to annotate their web sites using RDFa or Microdata. When this happens, hopefully in a near future, it will definitely be a valuable resource for the information extraction community, but until then, we can either annotate our own dataset or use HAREM. In this work, we will describe a way to easily annotate our own dataset, with a set of entity types of our choice, based on Brat.

2.2 Semantic Ranking and Search

Until now, we have been surveying information extraction techniques, with the final goal of automatically building a knowledge base to support a semantic ranking task. Knowledge bases can be used to empower search engines or to generically improve ranking, even in query-independent tasks. Next, we provide a bit of context regarding some of the central tasks of a semantic search engine that are representative of the integration of IE and IR. We cover query understanding, semantic matching, semantic search on text and knowledge bases and ontology-based ranking.

Query understanding involves conferring meaning to a keyword query. This is frequently done by segmenting the query and associating entities, types, or attributes to keyword chunks, through what is usually called semantic tagging [7]. Hu et al. [12] present a method for understanding query intent based on an index of Wikipedia concepts, supported on the idea of a graph of linked concepts. Given a query, the Wikipedia index is searched, returning the best matches to Wikipedia pages representing specific concepts. Whenever there is no match, the original query can be fed to a traditional *ad hoc* document retrieval search engine to build new queries from related text – in particular, the authors used Live Search[4], which has,

[3] `http://www.portaldalinguaportuguesa.org/?action=acordo&version=1990`
[4] `http://search.live.com`

at the time of this writing, been replaced with Bing[5]. The title and the description of the top-K results is used to emit additional queries to the Wikipedia index, giving more weight to the title. The final result is a ranked list of Wikipedia concepts that better illustrates query intent, covering not only concepts that are directly mentioned in the original query, but also concepts related to those, that are theoretically close in the graph of linked concepts. Their idea is not very different from the idea of community structure in a graph [20] – dense subgraphs, where member nodes have more connections between themselves than to the remainder of the network, usually represent similar or related concepts.

A bit different from query understanding, semantic matching [14] is concerned with solving mismatch issues between query and documents (e.g., searching for "ny times" should also match documents containing "New York Times"). While semantic matching is different from semantic search, some semantic matching solutions can still take advantage of techniques like named entity recognition, or knowledge bases like ConceptNet [11], for query expansion. Nevertheless, semantic search on text and knowledge bases [2] is the main stage for the combination of information extraction and information retrieval. In particular, the two areas meet when doing keyword search or question answering on combined data (text + knowledge bases). While semantic search can exist solely over knowledge bases, it can seldom exist solely over text, even though some semantics can be captured from text with techniques like word embeddings (e.g., word2vec [15]). Thus, information extraction plays a central role in search engines of the future, already supporting well-known search engines like Google, through knowledge bases like Knowledge Graph[6].

Finally, there is also work by Vallet et al. [24] showcasing the potential of ontology-based ranking, where a corpus is already linked to instances in a knowledge base. They associate a weight to each annotation, inspired by TF-IDF, but applied to knowledge base instances as opposed to terms, over a text document. The approach is to convert the user query into an RDQL query (a SPARQL predecessor), which is used to retrieve tuples from the knowledge base. Documents annotated with the instances from the obtained tuples are then retrieved as well and presented to the user sorted by a ranking function that measures semantic similarity. Semantic similarity is computed based on the cosine similarity between a query vector $\vec{q}$ and a document vector $\vec{d}$. The document vector $\vec{d}$ is defined by the analogous TF-IDF weights of the instances in the documents, while the query vector $\vec{q}$ is defined by weights assigned to the variables in the RDQL query, which can be either selected by the user, or obtained through personalization or frequency analysis.

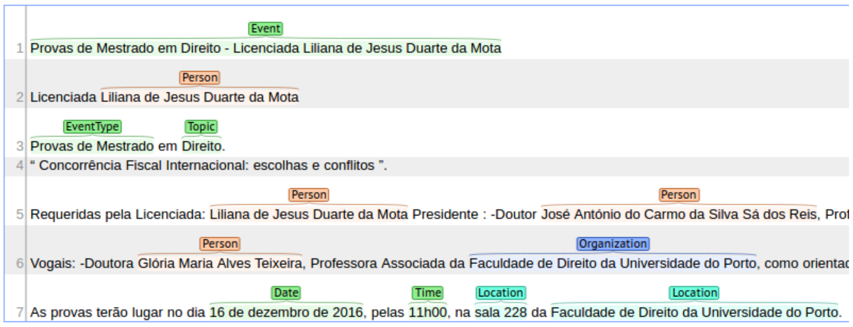
2.2.1 Our Approach in Context

Despite different from semantic search, in the sense that there is no issued query, the problem of semantic ranking we present here is closely related to the approaches we covered so far. We rank academic events based on the news articles that announce them. In particular, we use an information extraction pipeline for named entity recognition, as well as relation extraction, over institutional news. We then build a knowledge base, containing event information, but also *partOf* relations between pairs of organizations and locations. This enables us to rank events based on historical information on entity popularity and news article clicks, while also propagating popularity through *partOf* relations. It means that our ranking function captures the following information: if events about "Information Retrieval" are popular and

[5] `http://www.bing.com`
[6] `https://www.google.com/intl/es419/insidesearch/features/search/knowledge.html`

Table 1 Entity types used to annotate the SIGARRA News Corpus, along with examples aligned with Figure 2. Listed entity types were used to populate the [**entities**] section of the `annotations.conf` file in Brat.

Entity Type	**Example**
Person	*Liliana de Jesus Duarte da Mota*
Organization	*Faculdade de Direito da Universidade do Porto*
Event	*Provas de Mestrado em Direito - Licenciada Liliana de Jesus Duarte da Mota*
EventType	*Provas de Mestrado*
Topic	*Direito*
Location	*sala 228*
Date	*16 de dezembro de 2016*
Time	*11h00*



Provas de Mestrado em Direito - Licenciada Liliana de Jesus Duarte da Mota
Licenciada Liliana de Jesus Duarte da Mota
Provas de Mestrado em Direito.
“ Concorrência Fiscal Internacional: escolhas e conflitos ”.
Requeridas pela Licenciada: Liliana de Jesus Duarte da Mota Presidente : -Doutor José António do Carmo da Silva Sá dos Reis, Prof
Vogais: -Doutora Glória Maria Alves Teixeira, Professora Associada da Faculdade de Direito da Universidade do Porto, como orientac
As provas terão lugar no dia 16 de dezembro de 2016, pelas 11h00, na sala 228 da Faculdade de Direito da Universidade do Porto.

Figure 2 Example of an annotated fragment of text, for a SIGARRA news, in the Brat rapid annotation tool.

events at “Faculdade de Engenharia da Universidade do Porto” are also popular, then an event at “Departamento de Engenharia Informática” about “Information Retrieval” will have a high probability of being relevant to the community and should have a higher rank. While the link to “Information Retrieval” was directly established, “Departamento de Engenharia Informática” indirectly received a high popularity weight because it is *partOf* “Faculdade de Engenharia da Universidade do Porto”.

Since our system does not do entity disambiguation, whenever the same department naming is used across different universities, popularity will be propagated through all *partOf* links. In order to improve this, entity disambiguation is, in fact, one solution, but we can also assign an uncertainty weight proportional to the number of target nodes in the *partOf* relation, similar to IDF – the more target nodes are reached, the less reliable is the information. For our particular domain, however, we predict this to be of low impact – in fact, considering this type of relations without disambiguation might be a way of avoiding popular venues to control the ranks, giving a chance of similar events, happening at different venues, receiving a higher rank.

3 Data Acquisition, Model Training and Evaluation

The ANT search engine periodically collects and processes data from SIGARRA, the information system at the University of Porto. Data is retrieved using a combination of XPath and CSS selectors and stored in a relational database. While most of the data, like student or staff profiles, is structured, there is also some unstructured data from SIGARRA news.

Table 2 Evaluation of the named entity recognition module, based on the test set for the SIGARRA News Corpus.

Avg. Precision	Avg. Recall	Macro F1
0.633220	0.371867	0.468563

In this work, we are particularly interested in extracting information about events, locked inside SIGARRA news as natural language text. SIGARRA news announcing events can be identified based on whether there is an associated event start or end date in the database. We only retrieve and process news articles with at least one event date associated and skip articles in languages other than Portuguese. Steps from data acquisition to model training and evaluation are as follows:

1. Query the relational database for a subset of news articles that announce events and store them as a CSV file (one article per row).
   We retrieved the news articles for the 1,000 *closest upcoming events, starting from December 23, 2016, and containing contributions from all faculties and organic units of the University of Porto.*
2. For each row in the CSV, prepare a `txt` file containing title, subtitle and content, as well as an empty `ann` file.
   SIGARRA news articles communicate a lot of relevant information in the title and subtitle. In particular, we used the title to build `Event` *entities and the subtitle to extract information like the location or the date of the event. The content was provided as an HTML fragment that we converted into text using* `bs4.BeautifulSoup`*, after removing* `script` *and* `style` *tags.*
3. Put individual files within a subdirectory in the `data/` directory of the Brat rapid annotation tool and create an `annotations.conf` file with the list of entity types to annotate (shown in Table 1, along with examples).
4. Run `./standalone.py` in Brat's root directory and manually annotate the corpus, as shown in Figure 2 (we annotated 25 out of the 1,000 retrieved documents).
5. Split the annotated corpus into two separate directories, one for training (70%; 18 news articles) and another one for testing (30%; 7 news articles).
   While we admit the size of the annotated corpus is not ideal, surprisingly 18 annotated documents were enough to build a working system. Additionally, the annotated corpus can be extended over time, in order to retrain and re-evaluate the system.
6. Convert training documents into a single `col` file (tab-separated format supported by Stanford NER), and testing documents into individual `col` files to enable per-document evaluation.
7. Train a Conditional Random Field (CRF) using Stanford NER command line interface and obtain a model for named entity recognition (NER).
8. Evaluate the NER module. Extract entities from the original, non-annotated documents of the test set, using `StanfordNERTagger` from NLTK along with the learned CRF model. Compare extracted entities with annotated entities based on the `col` files, computing metrics like precision, recall and F-score.
   There are several evaluation methods for NER [17]. In this particular case, we used the CoNLL evaluation scheme, where only exact matches are considered as true positives. We calculate the precision and recall for each document and then the overall averages for the test set. We also compute the macro-averaged F1-score.

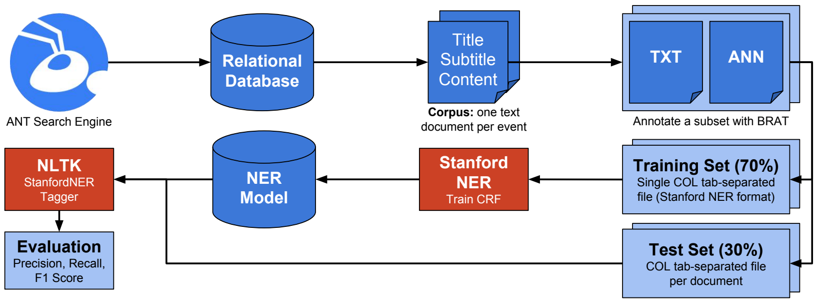

ANT Search Engine
Relational Database
Title
Subtitle
Content
Corpus: one text document per event
TXT
ANN
Annotate a subset with BRAT
NLTK
StanfordNER Tagger
NER Model
Stanford NER
Train CRF
Training Set (70%)
Single COL tab-separated file (Stanford NER format)
Evaluation
Precision, Recall, F1 Score
Test Set (30%)
COL tab-separated file per document

Figure 3 Data acquisition, named entity recognition and evaluation. Data is obtained from the ANT search engine database and pre-processed to enable CRF training and testing.

To summarize, Figure 3 presents a complete overview of the named entity recognition module, from data acquisition to evaluation. As we can see, starting from the ANT search engine, we access the relational database, in order to obtain a corpus, initially stored as a CSV file. We then convert each document into a text file, with an associated empty annotations file. The corpus is then annotated using Brat rapid annotation tool and split into a training set and a test set. The training set is used to train a CRF[7], obtaining a NER model. The NER model is then used by the `StanfordNERTagger` from NLTK to evaluate the effectiveness of the named entity recognition module based on the test set. Evaluation results are shown in Table 2 – we obtained an average precision of 63%, which is acceptable given the reduced size of the annotated corpus, as well an average recall of 37% and a macro-averaged F1-score of 47%.

4 Information Extraction

When running the information extraction pipeline, we simply query the relational database, iteratively processing each individual news article, until a set of triples with the identified entities and relations in the document is constructed and loaded into the triple store. The pipeline we defined is mostly based on NLTK and includes the following steps, that are applied to each text sequentially:

detect_language() Our pipeline was trained using a Portuguese corpus. In order to ensure we only process Portuguese documents, we use the Python wrapper for the well-known `langdetect` Java library[8]. Whenever a text is not identified as ‘pt’, it is simply skipped. The associated event might still be featured in ANT’s event widget, but only when it’s extremely close to the date and no other, more relevant events, are held simultaneously.

segment_sentences() The first step in processing a SIGARRA news article is to split the text into individual sentences. We used the pre-trained Portuguese model for the Punkt sentence tokenizer provided by NLTK. Challenges associated with this task are frequently associated with distinguishing actual sentence ends from periods that do not end a sentence (e.g., ‘Mr.’ or ‘Dr.’). While this is a task that is essentially solved and already achieves a high precision in the state of the art, given the quality of our text, many times sentences were incorrectly identified. However, we did not find this to be particularly

[7] We used a default configuration for Stanford NER, based on the example in the FAQ: `http://nlp.stanford.edu/software/crf-faq.shtml#a`.
[8] `https://github.com/Mimino666/langdetect`

impactful in the end result, as we did not rely too much on syntactic or dependency parsing or even on POS tags, which we only used for relation extraction.

tokenize_sentences() Given a list of strings, obtained from the previous step and corresponding to a sentence each, we split each string into individual words, as well as punctuation. For this, we used `WordPunctTokenizer` from NLTK, instead of the default tokenizer implemented in `nltk.tokenize.word_tokenize`, which was `TreebankWordTokenizer`. While this tokenizer reliably identified words, without splitting for instance compound words by dashes, it did not implement the `span_tokenize()` method, which was fundamental to simplify the conversion process from the Brat standoff format `ann` files into the Stanford NER tab-separated format `col` files. During tokenization, we also replaced any slash characters by a dash, since slashes are removed by `StanfordNERTagger` and, as we will explain next, we will need to match tagged sentences, word by word, in order to build a tree combining POS tags and named entity tags.

pos_tag_sentences() Given a list of lists of words and punctuation, from the previous step, we assigned a POS tag to each word, obtaining a list of lists of *(word, pos_tag)* tuples. Since there is no pre-trained model for POS tagging Portuguese sentences in NLTK, we used the provided Floresta treebank corpus to learn our own model. For training, we used a `nltk.BigramTagger`, falling back to a `nltk.UnigramTagger` and then to a `nltk.DefaultTagger` which always identifies a word as a noun (the most common), in case all else fails. An evaluation of a similar POS tagger based on Floresta treebank was already available[9], showing an accuracy of 89%, 87% and 18%, respectively for each tagger.

ne_tag_sentences() Given a list of lists of words and punctuation (without POS tags), we assigned a named entity to each word, obtaining a list of lists of *(word, ne_tag)* tuples. In order to do this, we used the model we had trained for Stanford NER, feeding it to the `StanfordNERTagger` (see step 8 in Section 3 for more details).

build_sentence_trees() Given the lists generated by `pos_tag_sentences()` and `ne_tag_-sentences()`, we generate a list of `nltk.tree.Tree` (one per sentence). In order to do this, we use the `nltk.chunk.util.conlltags2tree()` function, which takes a sentence argument as a list of *(word, pos_tag, ne_tag)* tuples. In order to build such tuples, we assume that an exact equivalence can be established between the two lists, otherwise the system will fail, raising an exception, which results in the skipping of the news article being processed. The resulting tree has three levels, a root level (the sentence), a mid-level (the entity types) and a bottom-level (leaves corresponding to chunks of words belonging to a named entity, as aggregated by the mid-level nodes).

extract_entities() In order to extract entities, we simply iterate over the trees constructed in the previous step, aggregating tokens per entity type. This is one of the outputs of our system, which is mainly used for evaluation purposes, regarding the effectiveness of the named entity recognition module. Results are saved as a human-readable `ent` text file (one per document), containing lists of entities, grouped by type, with the frequency of the entity in the document. We also save the same information as a `col` file, in order to compare predicted entities with the ground truth established in the test set. Results have already been shown in Table 2 and commented at the end of Section 3.

extract_relations() Relation extraction consists of identifying links between pairs of entities. In order to do this, we defined a set of rules, using regular expressions to identify relations between two types of entities, based on `nltk.sem.extract_rels()`. We then defined

[9] `http://www.nltk.org/howto/portuguese_en.html`

an associated list of rules to map the extracted relations into one or more triples in the knowledge base, possibly in reverse order. For example, the rule *('Location', '(da|do)/n', 'Location')* was used to identify *partOf* relations between two locations. Each Location entity was then mapped to a `dul:Place` class, from the DOLCE+DnS Ultralite ontology (DUL), and the *partOf* relation was mapped to the `dul:isPartOf` property from the same ontology. Event information was modeled using the Linked Open Descriptions of Events ontology (LODE), which focuses on defining a `lode:Event` class and a set of properties like `lode:atPlace`, that links to `dul:Place`, or `lode:involvedAgent`, that links to `dul:Organization` or `dul:Person`. The Time ontology was also used to describe date and time, using `lode:atTime` to link to a `time:TemporalEntity` and, in particular, a `time:Instant` with the property `time:inXSDDateTime` linking to a `xsd:dateTime` literal.

build_default_relations() Since many of the extracted entities were not featured in any relation, and in order to expand our knowledge base to better support the ranking task, we decided to generate some default triples that linked identified entities to the corresponding `lode:Event`. With this step, we compromised the quality of the information, in the sense that we considered for instance all dates as part of the `lode:atTime` relation and we know that some dates were in fact deadlines associated with events. The same is true for locations, as different places were sometimes referenced in news articles, besides the venue of the event. This is something that we intend to improve over time, either by using different properties like `dul:associatedWith`, or by improving automatic relation extraction.

load_relations_into_virtuoso() Finally, we generate an N-Triples (`nt`) file with the relations identified for each document, including relations of type *isA*, which annotate each entity with its class. This `nt` file is then loaded into OpenLink Virtuoso, through a POST request to the `/sparql-graph-crud-auth` endpoint[10], storing this information in a separate `ant:EventsKnowledgeBase` graph, using the same naming convention for events and their associated news article, already part of an existing ANT ontology[11].

5 Event Ranking

Loading the extracted information, in the form of triples, into the search engine's knowledge base, completed the process of annotating existing SIGARRA news articles. This included links to the extracted entities, as well as the capturing of interesting relations, like *partOf*, that were previously locked inside the news body as natural language. Combining this information can confer a degree of knowledge to our search engine and, in particular, it can be applied to the task of event ranking.

Prior to the approach we present here, ANT simply displayed the three closest upcoming events, without any particular ranking strategy. Equipped with a knowledge base we can do much more. In particular, we propose two scores based on entities associated with events: the entity popularity score, $score_{pop}(e, E_e)$, and the entity click score, $score_{clk}(e, E_e)$. The entity popularity score is based on the popularity of individual entities, as defined by the frequency an entity appears in distinct news articles over the whole corpus. The entity click score is similar to the popularity score, but is limited to the entities that are linked to clicked

[10] `https://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/VirtGraphProtocolCURLExamples`
[11] Example of URI for `lode:Event` / `ant:NewsArticle`: `http://infolab.fe.up.pt/ontologies/ant#Sigarra_News_Article_53369_From_Faculdade_de_Engenharia_da_Universidade_do_Porto`.

Listing 1 SPARQL query to compute $score_{clk}(e, E_e)$ based on `lode:`.

```
1  SELECT ?event ?code ?school (SUM(?count) AS ?score)
2  WHERE {
3    {
4      SELECT ?agent (COUNT(?agent) AS ?count)
5      FROM ant:EventsKnowledgeBase
6      WHERE {
7        ?event a lode:Event .
8        ?event ant:wasClicked "true"^^xsd:boolean .
9        ?event lode:involvedAgent ?agent .
10     }
11     GROUP BY ?agent
12   }
13   ?event a lode:Event .
14   ?event lode:involvedAgent ?involved_agent .
15   ?agent dul:partOf* ?involved_agent .
16   OPTIONAL {
17     ?event ant:hasCode ?code .
18     ?event ant:hasFaculty ?school .
19   }
20 }
21 GROUP BY ?event ?code ?school
```

event news – this click status is transfered to the knowledge base daily, when the IE pipeline is ran, and is stored using the `ant:wasClicked` property, linking to a `xsd:boolean` literal.

The two scores are computed using a SPARQL query and then stored in the relational database, each in their own column of the news articles table. Events are then retrieved by combining the number of days remaining to the event start date and the entity popularity and click scores. Listing 1 shows the SPARQL query used to calculate $score_{clk}(e, E_e)$ for all `lode:Event` instances in the system. The $score_{pop}(e, E_e)$ is calculated using a similar query, where we remove the statement in line 8, discarding the constraint for clicked events. As we can see, each entity score is calculated, per event, based on the total number of links to the entities that are associated with the event. We illustrate this by using `lode:involvedAgent` property for classes `dul:Person` and `dul:Organization`.

After computing and storing the two entity scores for each event, we calculate the final score, for event ranking, as shown in Equation 1. Given event e and entities E_e, associated with event e, the final score is calculated based on a weighted average of three factors: days to event, ΔT_e, entity popularity score, $score_{pop}(e, E_e)$, and entity click score, $score_{clk}(e, E_e)$.

$$score(e, E_e) = w_1 \times \frac{1}{\Delta T_e + 1} + w_2 \times \frac{score_{pop}(e, E_e)}{\max_e\{score_{pop}(e, E_e)\}} + w_3 \times \frac{score_{clk}(e, E_e)}{\max_e\{score_{clk}(e, E_e)\}} \quad (1)$$

We have deployed this ranking strategy in the ANT search engine, manually tuning the weights in order to prioritize the number of days to the event, since close events are more relevant, then considering the implicit feedback based on the entity click score and only then the entity popularity score. The version we deployed only relies on entities of type Person and Organization and uses $w_1 = 0.5$, $w_2 = 0.2$ and $w_3 = 0.3$. We will, in the future, measure the impact of this change in ranking strategy through implicit feedback from news event clicks, in particular looking for an increase in the number of clicks, and we will also experiment with the automatic tuning of the weights.

6 Conclusions

We presented a complete implementation of an information extraction pipeline, from data acquisition to knowledge base population. During this process, we covered different approaches, either based on machine learning or handcrafted rules. We took advantage of corpora that was directly integrated into the NLTK library, but also of our own manually annotated documents. We also built regular expressions for relation extraction, and ontology mapping rules for building triples to load into a knowledge base. We then described a practical application of the extracted information, to the area of information retrieval, where we tackled the problem of query-independent ranking for a corpus of news articles announcing events. In particular, we proposed a temporal ranking approach combined with automatically compiled knowledge about the events being ranked. Through this, we showcased how *partOf* relations can be harnessed to influence the ranking of documents beyond directly mentioned entities, by taking advantage of related entities, mentioned in different documents.

The future of search engine intelligence highly depends on the unified efforts of information extraction and information retrieval. While we already have high quality machine learning techniques to support search by modeling "thought through numbers", we still lack the ability to effectively model "thought through language" in order to build search engines that can better assist users, not only by better understanding them, but also by helping them sort through a large amount of information locked within textual documents in natural language.

References

1 Michele Banko, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. Open information extraction from the web. In *20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2670–2676, 2007.

2 Hannah Bast, Björn Buchhold, and Elmar Haussmann. Semantic search on text and knowledge bases. *Foundations and Trends in Information Retrieval*, 10(2–3):119–271, 2016.

3 Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly, 2009.

4 Nuno Cardoso. REMBRANDT - reconhecimento de entidades mencionadas baseado em relações e análise detalhada do texto. In Cristina Mota and Diana Santos, editors, *Desafios na avaliação conjunta do reconhecimento de entidades mencionadas: O Segundo HAREM*, pages 195–211. Linguateca, 2008.

5 Hamish Cunningham, Valentin Tablan, Ian Roberts, Mark A Greenwood, and Niraj Aswani. Information extraction and semantic annotation for multi-paradigm information management. In Mihai Lupu, Katja Mayer, Noriko Kando, and Anthony Trippe, editors, *Current Challenges in Patent Information Retrieval*, pages 307–327. Springer, 2011.

6 Hamish Cunningham, Yorick Wilks, and Robert Gaizauskas. GATE: a general architecture for text engineering. In *16th Conference on Computational Linguistics*, pages 1057–1060, 1996.

7 José Devezas and Sérgio Nunes. Index-based semantic tagging for efficient query interpretation. In *International Conference of the Evaluation Forum (CLEF)*, pages 208–213, 2016.

8 Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *20th International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 601–610, 2014.

9 Erick Rocha Fonseca and João Luís G. Rosa. Mac-morpho revisited: Towards robust part-of-speech tagging. In *9th Brazilian Symposium in Information and Human Language Technology*, pages 98–107, 2013.

10 Cláudia Freitas and Susana Afonso. *Bíblia Florestal: Um manual lingüístico da Floresta Sintá(c)tica.* Linguateca, 2007.

11 Catherine Havasi, Robert Speer, and Jason Alonso. ConceptNet 3: a flexible, multilingual semantic network for common sense knowledge. In *Recent Advances in Natural Language Processing (RANLP)*, pages 27–29, 2007.

12 Jian Hu, Gang Wang, Fred Lochovsky, Jian-tao Sun, and Zheng Chen. Understanding user's query intent with Wikipedia. In *18th International Conference on World Wide Web*, pages 471–480, 2009.

13 John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data. In *Eighteenth International Conference on Machine Learning (ICML)*, pages 282–289, 2001.

14 Hang Li and Jun Xu. Semantic matching in search. *Foundations and Trends in Information Retrieval*, 7(5):343–469, 2014.

15 Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

16 Cristina Mota and Diana Santos. *Desafios na avaliação conjunta do reconhecimento de entidades mencionadas: O Segundo HAREM.* Linguateca, 2008.

17 David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

18 Kamel Nebhi. Named entity disambiguation using freebase and syntactic parsing. In *First International Conference on Linked Data for Information Extraction*, pages 50–55, 2013.

19 Joakim Nivre, Johan Hall, and Jens Nilsson. MaltParser: A data-driven parser-generator for dependency parsing. In *The Fifth International Conference on Language Resources and Evaluation (LREC)*, pages 2216–2219, 2006.

20 Pascal Pons and Matthieu Latapy. Computing communities in large networks using random walks. *Journal of Graph Algorithms and Applications*, 10(2):191–218, 2006.

21 Ricardo Rodrigues, Hugo Gonçalo Oliveira, and Paulo Gomes. LemPORT: a high-accuracy cross-platform lemmatizer for portuguese. In *3rd Symposium on Languages, Applications and Technologies (SLATE)*, pages 267–274, 2014.

22 Wei Shen, Jianyong Wang, and Jiawei Han. Entity linking with a knowledge base: Issues, techniques, and solutions. *Transactions on Knowledge and Data Engineering*, 27(2):443–460, 2015.

23 Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. BRAT: a web-based tool for NLP-assisted text annotation. In *13th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 102–107, 2012.

24 David Vallet, Miriam Fernández, and Pablo Castells. An ontology-based information retrieval model. In *European Semantic Web Conference*, pages 455–470, 2005.